モンテカルロ法による待ち行列のシミュレーション

＜概　要＞
身近な事柄の中からテーマを見つけ、これをモデル化し、コンピュータでシミュレートする方法を学習する。ここでは乱数を使ったモンテカルロ法について学ぶ。
＜キーワード＞
乱数、モデル化、シミュレーション、モンテカルロ法、待ち行列

１．学習活動

（１）シミュレーション

シミュレーションとはどの様なことか、またシミュレーションを行うことによってどのような利益が得られるかを考える。（現実には起こりにくいことや危険なこと、長い時間を要するような事柄をコンピュータを用いて擬似的に発生させ、短時間で安全に、効率よく経過を観察できる）
課題：現在から、西暦３０００年１月１日午前０時までの太陽、地球、月の位置関係は正確にシミュレートできるが、空に浮かぶ綿雲の３０分後の位置と形のシミュレートはほとんど不可能である。なぜか考えてみよう。

（２）テーマとモデルの設定「待ち行列」

窓口にならぶ客とこれに応対する事務員の様子をモデル化し、様々な条件を与え状況の変化を観察する。

a　条件の選択と設定（モデル化）

モデル化を行うために必要な条件を挙げる。

- １分ごとに状況を区切って観察する。→「時系列的に観察を進める」
- １分間の間に客が来たかどうかを無作為に決定する。→「モンテカルロ法の利用」
- １人の客に対し事務員が行う業務に必要な時間を設定する。「処理時間の設定」
- 待合室の収容人数を設定する。→「待ち人数の最大値」
- 客がいないときは事務員は待機している。→「待機中の人件費」

経済的な側面から見たとき、帰っていく客が少ない方が良く、客をあまり多く待たせない範囲で事務員の能力は低い方がよい。（事務処理能力の高い事務員は人件費も高い）
都市部であれば待合室を広くすることはかなり費用がかかることも考えられる。
来客の頻度や事務員の人件費、客１人あたりの収益などから、どのような点を改善すればより利益を上げることができるか、コンピュータを使って何度も試行し最適解を得る。
計算結果として　総来客数、最大待ち人数、帰っていった客の人数、事務員待機時間を求めることにする。

b　事前準備

①乱数の発生

表計算ソフトでRAND（）という関数を使うと０～１までの乱数を発生させることができる。何回も発生させて試してみる。
Ａ列の最上段にRAND()を書き込み、列方向に１００～６００個コピーし、これをfrequency関数を用いて 0.1 刻みでヒストグラム化し、さらにグラフ化してみる。（図１、２）数多く乱数を発生させると、その分布はだんだん一様になっていくことがわかる。

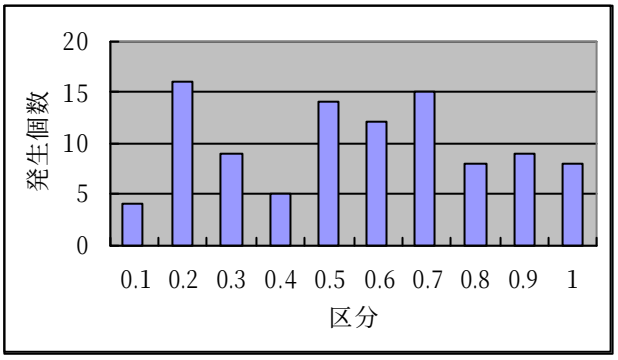

図１　１００個の乱数の分布

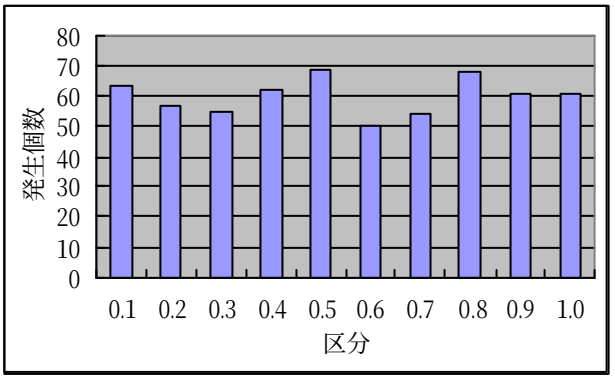

図２　６００個の乱数の分布

②モンテカルロ法による来客の有無のモデル化

物事の状態をモデル化する際、例えば投げ上げたボールの運動は2次方程式であらわすことができるが、これに対し乱数などを用いて、ある事柄が起きる起きないなどということをモデル化する場合を「**確率的方法**」といい、通常「**モンテカルロ法**」と呼ばれている。

1分間の間に客が来るか来ないかを無作為に決定するために(モデル化するために)モンテカルロ法を使う。

＜例＞80%の確率で客がやってくると思われる場合は、0～1までの乱数を発生させて0.8以上であれば「来ない」、未満であれば「来た」と判断すればよい。

この作業を数多く試行しその結果をまとめると、客がやってくる割合は限りなく80%に近くなる。

（3）計算の手順

この問題の場合について計算手順の概要を考える。

①来客確率、試行時間数、1人の客に要する処理時間、待合室の収容人数を設定

ここから繰り返す。

②乱数を発生させ客が来たか来ないかを来客確率と比較して決定する。

- **もし来客があった場合**
  来客総数を1増やす。待ち人数を1増やす。
- **もし待ち人数が待合室収容人数より多ければ**その客は帰る。帰った客の人数を1増やす。

③事務員の手が空いているか

- **もし空いていれば**待ち人数は何人か調べる。
  - **待ち人数が0のとき、**事務員待機時間の記録に1を加える。
  - **待ち人数が1以上のとき、**待ち人数を1人減らし、事務処理を1分間行う。
- **手が空いていないとき**（現在応対中）事務処理を1分間行う。

ここまでを試行時間数分繰り返す。

④最後に待合室に残った人の処理を行う。

（4）実際にやってみる

客が来る確率を0.5とすれば、コイン投げで乱数発生を代行できるので、クラスの仲間同士で、実際に客や事務員になって実行してみる。

（5）プログラミング

適当なプログラミング言語を使ってプログラミングを行う。

（6）プログラムの利用

1日の稼働時間を300分とし、来客確率が1分あたり0.5とし、事務員の処理能力と人件費の関係が次の表のようになっており、客1人あたりの利益が500円とし、待合室の収容人員に関わる費用を1人あたり100円/日としたとき、どのレベルの事務員を使い待合室の収容人数を何人にするのがもっとも経済的か。

事務員のレベル	所要時間	人件費
A	1分/人	8000円/日
B	2分/人	6000円/日
C	3分/人	4000円/日

このほか様々な条件設定をして、変化を試してみる。

（7）プログラム例

```
! モンテカルロ法の応用　待ち行列のシミュレーション
! WINDOWS版10進Basic（JIS Full BASIC準拠）により記述

!設定条件の入力
input prompt "1分あたりの来客確率（0～1の小数）=":来客確率
input prompt "試行する時間数（100～10000分の整数）=":時間数
input prompt "来客一人あたりの処理時間（1以上の整数）=":処理時間
input prompt "待合室収容人数（1～20人の整数）=":収容人数

!乱数の初期化
!randomize　　! !randomize の「!」をはずすと毎回違ったパターンの乱数を発生する

!試行の繰り返し開始
for　経過時間=1 to 時間数
   if rnd <= 来客確率 then
   !乱数を発生して来客有無を決める
      let 来客有無 = 1
   !来客があるときは1(無いときは0)
```

```
      let 来客総数 = 来客総数+1
 !来客の総数を記録するためのカウンター
      let 待ち人数 = 待ち人数+1
 !待ち人数を１人増やす
      if 待ち人数> 収容人数 THEN
 !収容人数より多いときは客が帰る
         let 待ち人数 = 待ち人数 - 1
         let 帰り人数 = 帰り人数 + 1
      end if
      if 待ち人数 > 最大待ち人数 then
 !最大待ち人数の記録更新
         let 最大待ち人数=待ち人数
      end if
   else
      let  来客有無= 0
   end if
   if 事務残=0 then
 !事務員の手が空いているとき
      if 待ち人数=0 then
!待っている客がいないとき
       let 事務員待機= 事務員待機 +1
!事務員の空き時間として記録
      else
         let 待ち人数= 待ち人数-1
 !待っている客がいれば客を受け付ける
         let 事務残 = 処理時間-1
 !新しい客に対する事務を１分こなす
      end if
   else
!事務員の手はふさがっている（客に対応中）
      let  事務残=事務残-1
!事務員は先ほどからの客に対する事務を１分こなす
   end if
   !   print "経過時間＝";経過時間
！以下のプログラム行の先頭の！をはずすと途中経過を観察できる
   !  print "来客有無＝"; 来客有無
   !   print "来客総数＝";来客総数
   !   print "待ち人数＝";待ち人数
   !   print "事務残り=";事務残
   !   print "帰り人数=";帰り人数
   !   print "事務員待機=";事務員待機
   !   input prompt " OK?":dumm$
next 経過時間
!試行ここまで
print
print "経過時間＝";経過時間 - 1
print "来客総数＝";来客総数
print "帰り人数=";帰り人数
print "受付済み人数＝";来客総数 - 帰り人数
print "事務員待機=";事務員待機
print "最大待ち人数＝";最大待ち人数
END
```

出力例

＜条件の設定＞

１分あたりの来客確率（0~1の小数）＝.5

試行する時間数（100~1000分の整数)＝300

来客一人あたりの処理時間(1以上の整数)＝3

待合室収容人数（1~20人の整数）＝10

```
経過時間＝  300
来客総数＝  145
帰り人数= 36
受付済み人数＝  109
事務員待機= 0
最大待ち人数＝  10
```

2　備　考

ソースコードはこのまま実行することができる。

ここで使用したＢＡＳＩＣ言語は次のサイトで入手できる。（無料）

http://www.vector.co.jp/authors/VA008683/

3　利用資料

特になし

4　参考資料

下記のホームページで筑波大学の大山崇先生による待ち行列シミュレーション例を見ることができます。

http://infoshako.sk.tsukuba.ac.jp/~tohyama/queue/mati.html